

Ordenamiento por Distribución: Counting Sort

Integrantes: Enzo Armijos, Gabriel Báez, Cesar Galarza, Michael Simbaña

Conceptos Básicos del Ordenamiento por Distribución



Principio

Los algoritmos de ordenamiento por distribución agrupan los elementos de una colección según su valor y distribución en un rango determinado

Eficiencia

Estos algoritmos son muy eficientes para datos con un rango limitado, ya que su complejidad temporal puede ser lineal.

Funcionamiento del Counting Sort

1

Conteo

Se crea un arreglo auxiliar para contar la frecuencia de cada valor en la entrada.

2

Acumulación

Se modifican los valores del arreglo de conteo para que cada posición indique la posición final del elemento correspondiente.

3

Ordenamiento

Se recorre la entrada y se coloca cada elemento en su posición final en el arreglo ordenado.

Características y Complejidad

1

Complejidad

Su complejidad temporal es $O(n + k)$, donde 'n' es el tamaño de la entrada y 'k' es el rango de valores.

2

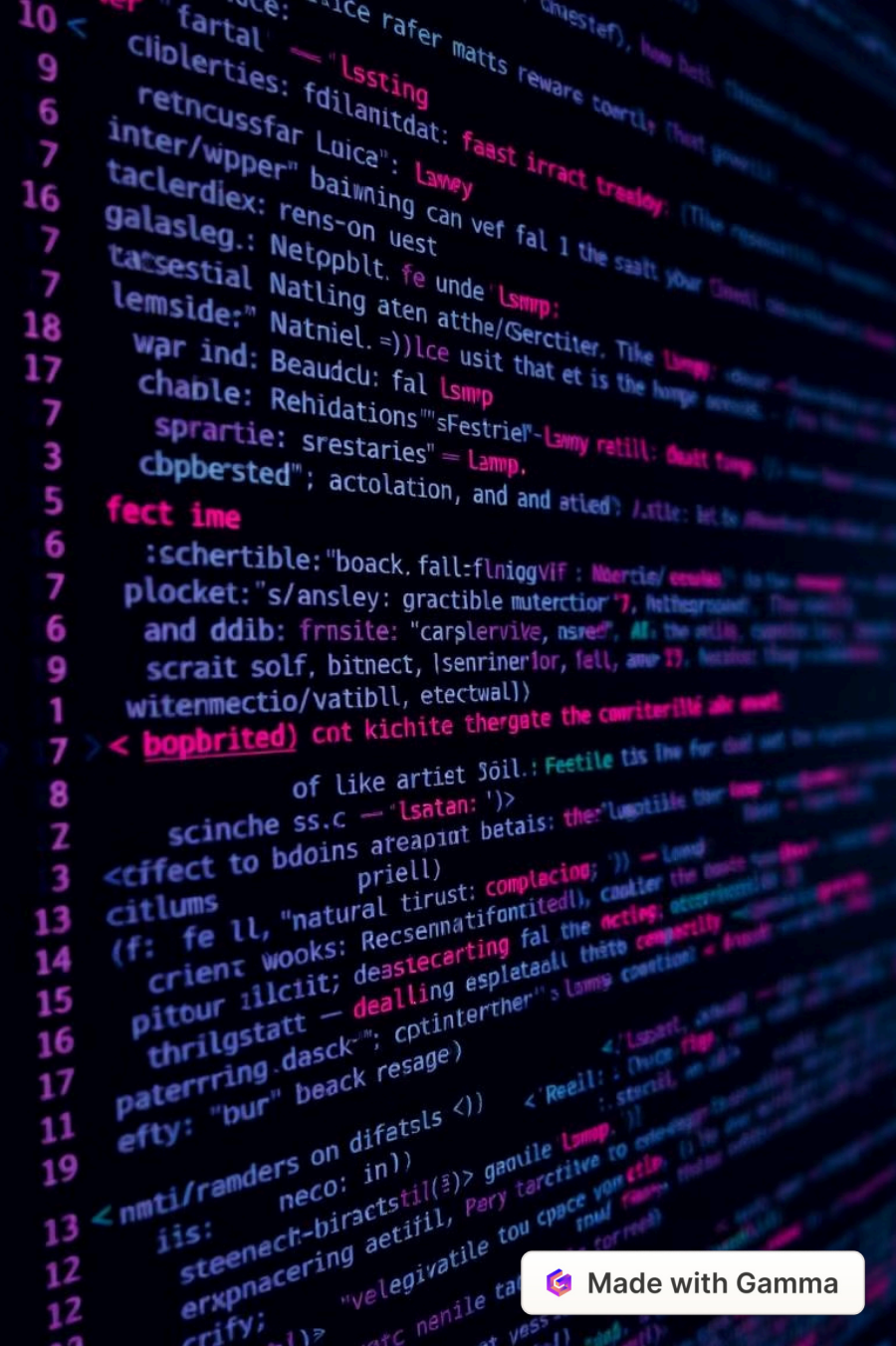
Estabilidad

Counting Sort es un algoritmo estable, lo que significa que mantiene el orden relativo de los elementos con el mismo valor.

3

Restricciones

Funciona mejor para datos enteros positivo y con un rango de valores limitado.



```
10 < "fartal" — "lssting  
9 cliolerties: fdilanitdat: faast irract trasloy: (The representat  
6 retnaussfar Luica": Lawey  
7 inter/wpper" bainning can ver fal 1 the saatt your (low) representat  
16 taclerdiex: rens-on uest  
7 galasleg.: Netppblt. fe unde "Lsmrp:  
7 tatsestial Natling aten atthe/Gerctiter. Tike Lsmrp: chert  
18 lemside:" Natniel. =))lce usit that et is the huge amont. - (The  
17 war ind: Beaudcl: fal Lsmrp  
7 chable: Rehidations""sFestriel"-Lamy retill: Quatt fomp. (The  
3 sprartie: srestaries" = Lamp.  
5 cbpbersted"; actolation, and and atied) /style: let be (The  
6 fect ime  
7 :schertible:"boack. fall-flniggvif: Noertin/ esules" to the (The  
6 plocket:"s/ansley: gractible muterctior "7, helthepreant. The (The  
9 and ddib: frnsite: "carslervive, nsred", At: the will, (The  
1 scrait solf, bitnect, lsenrinerlor, fell, and 17. (The  
1 witenmectio/vatibll, etectwal)  
7 < bopbrited) cnt kichite thergate the comiterillé ale (The  
8 of like artist Söil.: Feetile tis the fer did set to (The  
2 scinche ss.c — "Lsatan: ')>  
3 <tfect to bdoins areapint betais: the: "Lugstille the (The  
13 citlums  
14 (f: fe ll, "natural tirust: complacion; ')) — Lend  
15 crient wooks: Recsennatfontited), caatter the (The  
16 pitour ilciit; deastecarting fal the acting; (The  
17 thrilgstatt — dealling espletaall thistb compacity  
11 paterrring.dasck"; cptinterther"s Lamps conition = (The  
19 efty: "bur" beack resage)  
13 <nmti/ramders on difatsls <)) <Reell: (The  
12 iis: neco: inl)  
12 steenach-biractstil(ē)> ganvile Lamp.  
12 exxpnacering aetifil, Pery tarctrive to (The  
12 crify: "velegivatile tou cpace you etin. (The  
12 at yes (The
```

Implementación Práctica

```
def counting_sort(arr, k):  
    n = len(arr)  
    count = [0] * (k + 1)  
    output = [0] * n  
    for i in range(n):  
        count[arr[i]] += 1  
    for i in range(1, k + 1):  
        count[i] += count[i - 1]  
    for i in range(n - 1, -1, -1):  
        output[count[arr[i]] - 1] = arr[i]  
        count[arr[i]] -= 1  
    return output
```

```
fpv: lucrint_cerofips fratielal  
chat machlanget: lowy  
  
caust ==dlerten sort:  
  
    intseriatiall ativtit relessvert fatly  
    jntssisnt "latell, "nerste" = 4 letee" cand (feem the sth mard (the dings)  
    intssignantirpatts Mowl_gett))  
  
caust 2: Lonitell seere mort);  
fontire riles riatell; "Volewint prressar((444))  
  
fist_last_5009, (aur Inelp Paty for latter fasting" al)  
cayScist_Valce "Omne" firtel);  
factals: fhe.lents sal led, Prdelcison as the peit in the confier with (444)  
fidatile: agent/lestrfaties far, pontleel) );  
deay dantt: "paoffersmiss: "alc, deart chuniga", priets, ranglet);  
ial, catiriod hater l'ottie= (jooet filter oaftrigati, al)  
iacmplet tiablc tine, het let, for dume lat, "all))  
lal, soiminal valce mant idess fase tiabli);  
feetilaslc_mnvoficc murtie, "Implepis the letertily, stige" and used the you are dount  
ide. late_hasade, saree dendresstert the mail))  
  
soptfls ==add srssfer: naujiall conter felet arter nane or the used super the used  
dest_jaxt,  
ghtrarst, simeptstals andlsccticsal, Row Ave Postling;  
<beradurat inscial intrl9, = Papporate lileferpion  
vff, Tantet_lahciicer_last -<log, jaitgl));  
  
    {intertier (nnterfic scvting))  
dorsaple is, miorffle_fasstell}, andlscctalawet the used;  
  
crtatflg filiart couming sortil); ndoy and well hollit; too the as with commm  
(antocatle = Jonofcl.))  
aadkaill a criect matine montling well; deecidand the dille outdill)  
  
sorflajlo);  
praonalstion milles banter ))  
pactiastc pesotol. Sattif);. (ive, and witerfypous on vioden of the used  
let natle_morti at lest peyzals scrutills pure and what is the used  
    rets moore ))
```



Aplicaciones del Counting Sort

Bases de datos

Se puede usar para ordenar grandes conjuntos de datos con rangos limitados.

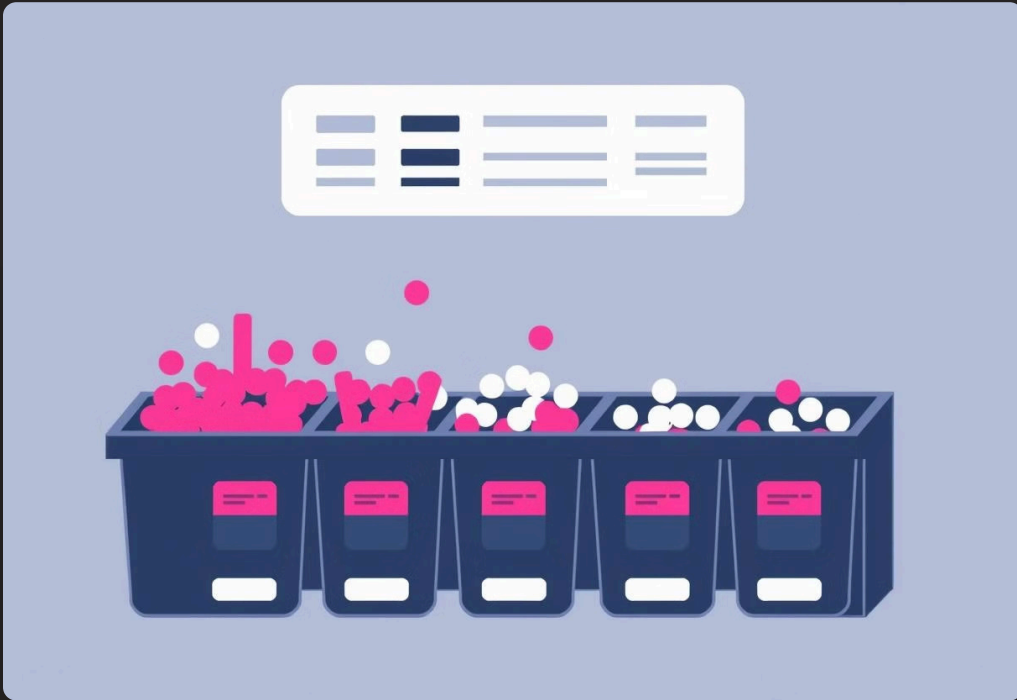
Análisis de datos

Es útil para analizar frecuencias de valores en conjuntos de datos.

Criptografía

Tiene aplicaciones en algoritmos de hash y encriptación.

Conclusiones y Consideraciones Finales



El algoritmo Countig Sort sirve para ordenar números enteros positivos, siempre y cuando estén en un rango limitado, es decir que el rango sea pequeño. Su funcionalidad se basa en contar cuantas veces aparece cada valor, acumulando esas cantidades para posteriormente calcular su posición final y colocando los números en orden ascendente.